AXIS BUSINESS SCHOOL

Session 2020-22

Master of Computer Application (MCA)

"Fake News Detection Project Using NLP & Machine learning"

SubmittedBy:

- Rohit Shukla
- ShivPratap
- Sudheer Kumar Gautam

GuidedBy:

Mr. Chandan Verma

H.O. D

Dr. Subha jain

Class Coordinator: Ms. Sadhana yadav

[FAKE NEWS DETECTION USING NLP & MACHINE LEARNING]

CONTENT

[FAKE NEWS DETECTION USING MACHINE LEARNING]

A Presentation by:

Rohit Shukla

Shivpratap

Sudheer Kumar Gautam

Introduction

Fake news has been around for decades and is not a new concept. However, the dawn of the social media age has aggravated the generation and circulation of fake news many folds. Fake news can be simply explained as a piece of article that is usually written for economic, personal, or political gains.

Social media is used for news reading, writing, and sharing. People are profiting by click bait and publishing fake news online. More clicks contribute to more money for content publishers.

Many scientists believe that fake news issues may be addressed by means of machine learning and artificial intelligence. Detection of such unrealistic news articles is possible by using various NLP techniques, Machine learning, and Artificial intelligence.

Major problem:

The growth of fake news on social media and the Internet is deceiving people to an extent that needs to be stopped. Fake News Can Affect Your Grades, harm your health, and makes Harder for People to See the Truth

Our Solution:

Our goal is to develop a reliable model that classifies a given news article as eitherFake or real.it can discriminate between “fake” and “true” news articles when it is trained with a certain dataset.

What is Fake News?

Fake news is false or misleading information presented as news whose source cannot be verified. It often has the aim of damaging the reputation of a person or entity or making money through advertising revenue or gain attention.Fake news stories usually spread through social media site like Facebook, Instagram, twitter and reddit.

Types of fake news

Clickbait. Content whose main purpose is to attract attention and encourage visitors to click on a link to a particular web page.

Satire/parody. A piece of writing, speech, or music that copies the style of somebody/something in a funny way.

Propaganda. Information and ideas that may be false or scam, which are used to gain support for a political leader, party, etc.

Biased: it means that preferring one group of people to another, and behave unfairly with them as a result.

Unreliable news: sources that don't always contain true, accurate, and up-to-date information.

Natural language processing (NLP)?

Natural language processing (NLP) is the ability of a computer program to understand human language as it is spoken and written -- referred to as natural language. It is a component of artificial intelligence (AI). There are two main phases to NLP: data preprocessing and algorithm development.

Data preprocessing involves preparing and "cleaning" text data for machines to be able to analyze it. Preprocessing involves Tokenization, Stop word removal, Lemmatization and stemming, Part-of-speech tagging. After this we use NLP algorithms. There many algorithms but we use mainly two

1. Rules-based system
2. Machine learning-based system.

TF-IDF

TF-IDF stands for "Term Frequency — Inverse Document Frequency". This is a technique to quantify words in a set of documents. We generally compute a score for each word to signify its importance in the document and corpus. This method is a widely used technique in Information Retrieval and Text Mining.

Confusion Matrix

A confusion matrix is a performance measurement technique for Machine learning classification. It is a kind of table which helps you to the know the performance of the classification model on a set of test data for that the true values are known.

Pickle

"Pickling" is the process whereby a Python object hierarchy is converted into a byte stream, and "unpickling" is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

ALGORITHMS

1) K-nearest neighbors (KNN)

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

Accuracy: 52%Prediction time: 77.70Learning time:3.93

2) Logistic regression

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of the target or dependent variable is dichotomous, which means there would be only two possible classes.

Accuracy: 95% Predication Time:0.33Learning Time: 11.88

3) Bagging classifier

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregates their individual predictions to form a final prediction.

Accuracy: 88% Predication Time: 0.62 Learning Time: 232.53

4) Naive Bayes Algorithm (NB)

Naive Bayes is a kind of classifier that uses the Bayes Theorem. It predicts membership probabilities for each class such as the probability that a given record or data point belongs to a particular class. The class with the highest probability is considered as the most likely class.

P(A|B) = P(B|A) * P(A) / P(B)

Accuracy: 91% Predication Time: 0.35 Learning Time: 1.11

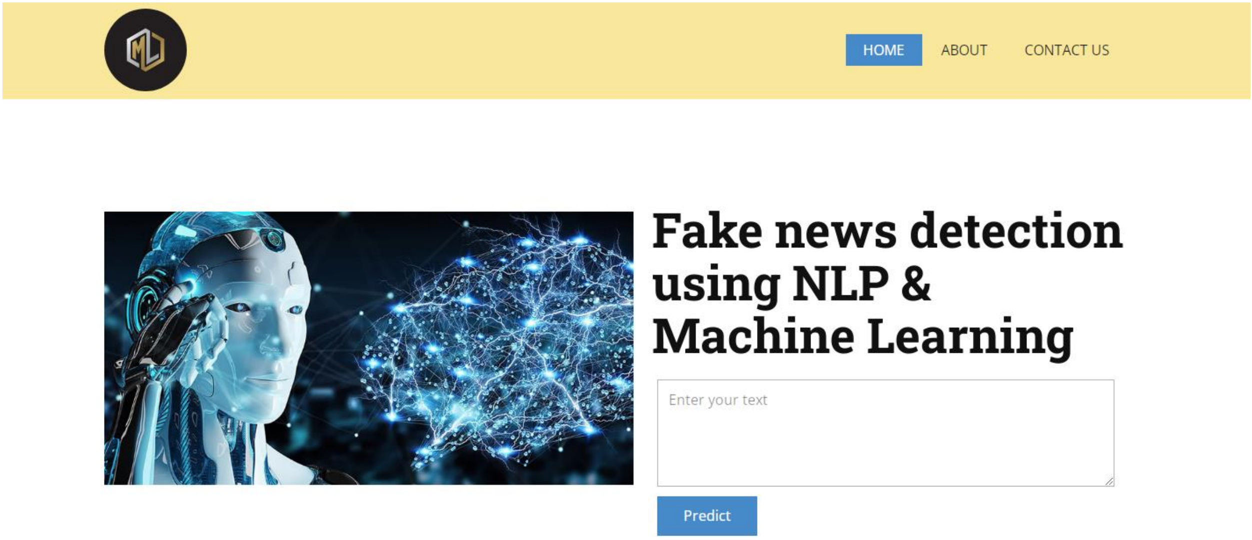
REQUIREMENTS:

- Python
- NumPy
- Pandas

- TF-IDF
- Itertools
- Matplotlib
- Scikit-Learn
- Spyder
- Heroku (For deployment)
- Flask
- NLP and Machine Learning Techniques

Explanation of model

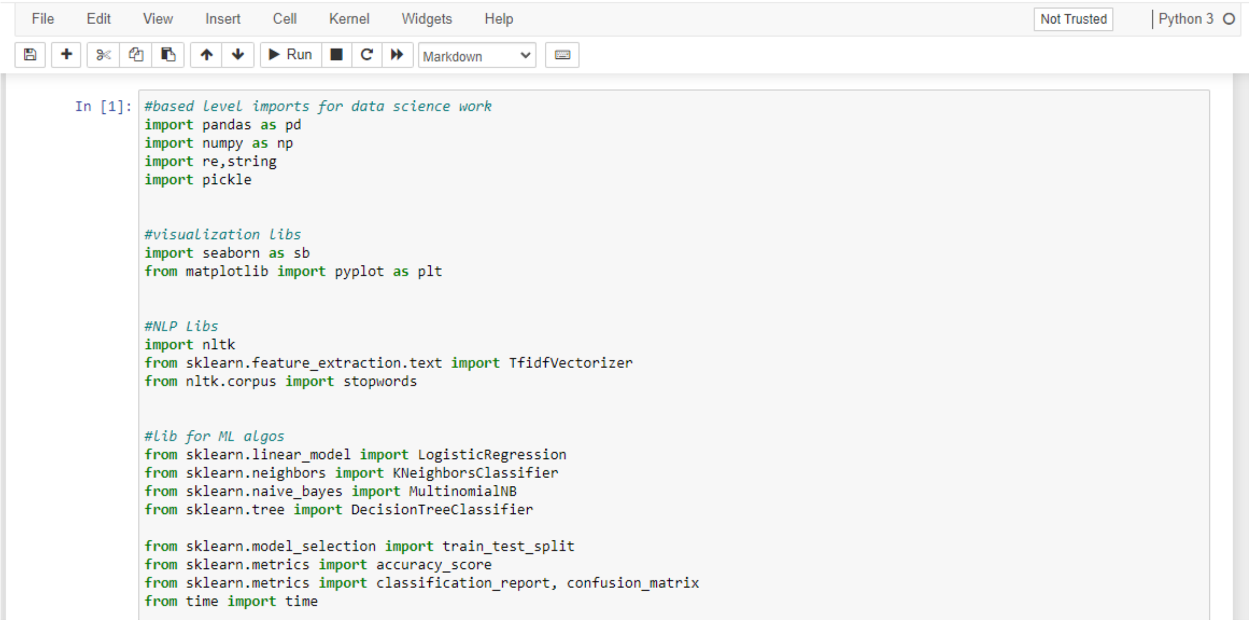
Front page



Fake news detection using NLP & Machine Learning

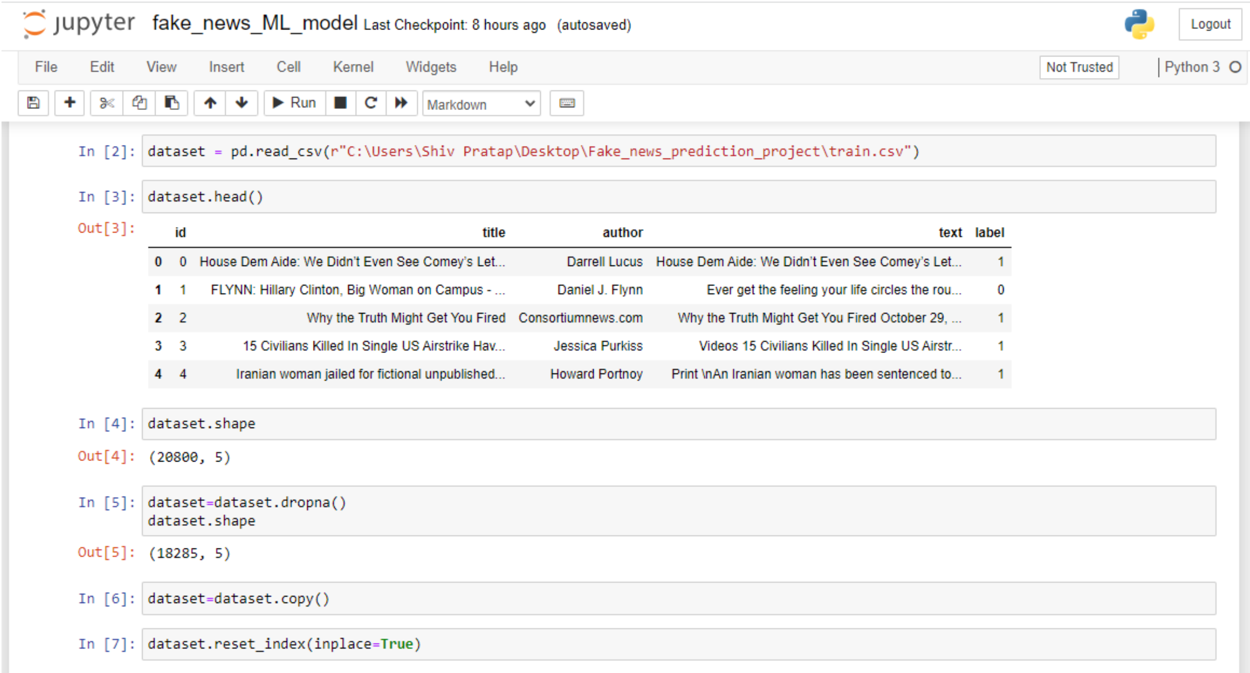
Fake News Detecting Model Code’s Using Machine Learning & NLP In JUPYTER Notebook

Importing the libraries



```
In [1]: #based level imports for data science work
        import pandas as pd
        import numpy as np
        import re,string
        import pickle


        #visualization libs
        import seaborn as sb
        from matplotlib import pyplot as plt


        #NLP Libs
        import nltk
        from sklearn.feature_extraction.text import TfidfVectorizer
        from nltk.corpus import stopwords


        #lib for ML algos
        from sklearn.linear_model import LogisticRegression
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.naive_bayes import MultinomialNB
        from sklearn.tree import DecisionTreeClassifier

        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score
        from sklearn.metrics import classification_report, confusion_matrix
        from time import time
```

Fetching the dataset (1 for unreliable and 0 for reliables)

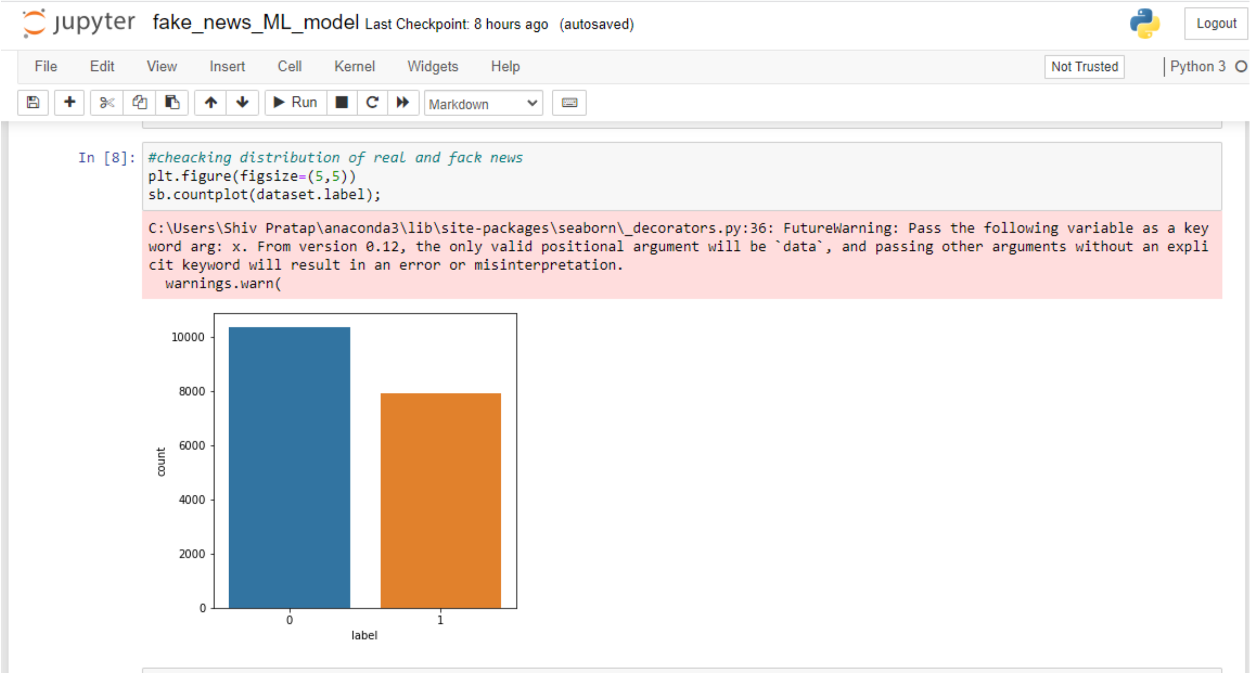

jupyter fake_news_ML_model Last Checkpoint: 8 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown

```
In [2]: dataset = pd.read_csv(r"C:\Users\Shiv Pratap\Desktop\Fake_news_prediction_project\train.csv")

In [3]: dataset.head()

Out[3]:
```

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

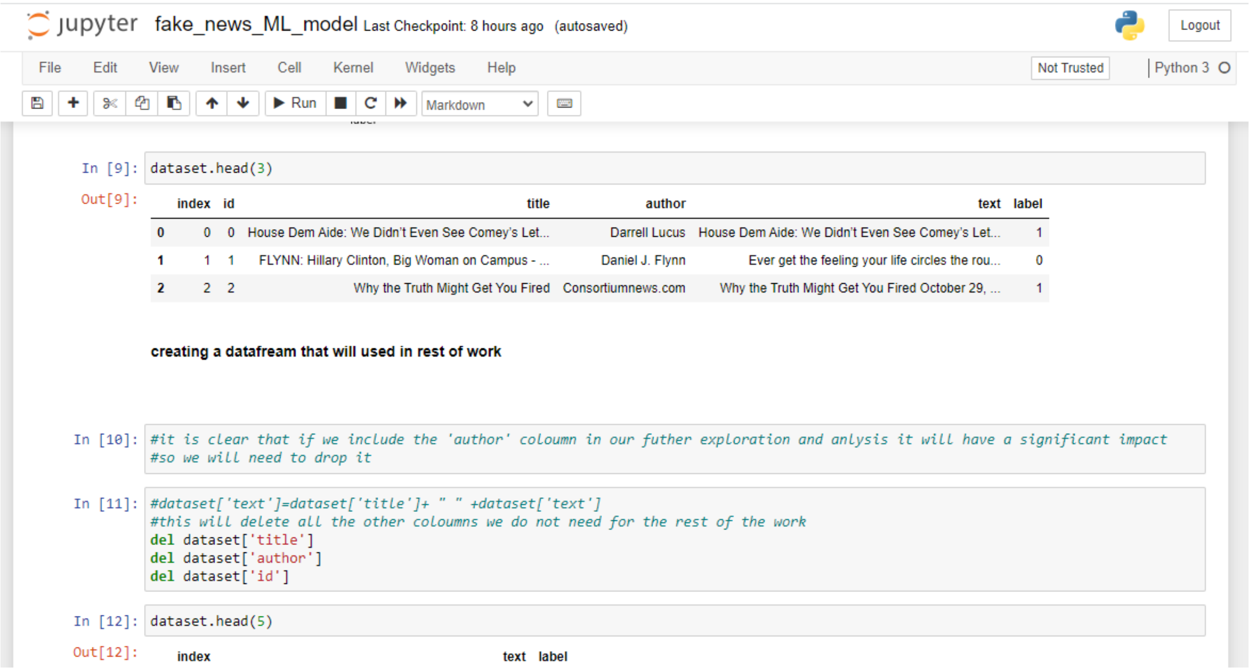
```
In [4]: dataset.shape

Out[4]: (20800, 5)

In [5]: dataset=dataset.dropna()
        dataset.shape

Out[5]: (18285, 5)

In [6]: dataset=dataset.copy()

In [7]: dataset.reset_index(inplace=True)
```

Fake news and real news distribution


jupyter fake_news_ML_model Last Checkpoint: 8 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
10000
8000
6000
4000
2000
0
count
0
1
label

```
In [8]: #cheacking distribution of real and fack news
        plt.figure(figsize=(5,5))
        sb.countplot(dataset.label);

C:\Users\Shiv Pratap\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a key
word arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Creating the DataFrame



In [9]:
```
dataset.head(3)
```

Out[9]:

	index	id	title	author	text	label
0	0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1

creating a datafream that will used in rest of work

In [10]:
```
#it is clear that if we include the 'author' coloumn in our futher exploration and anlysis it will have a significant impact
#so we will need to drop it
```

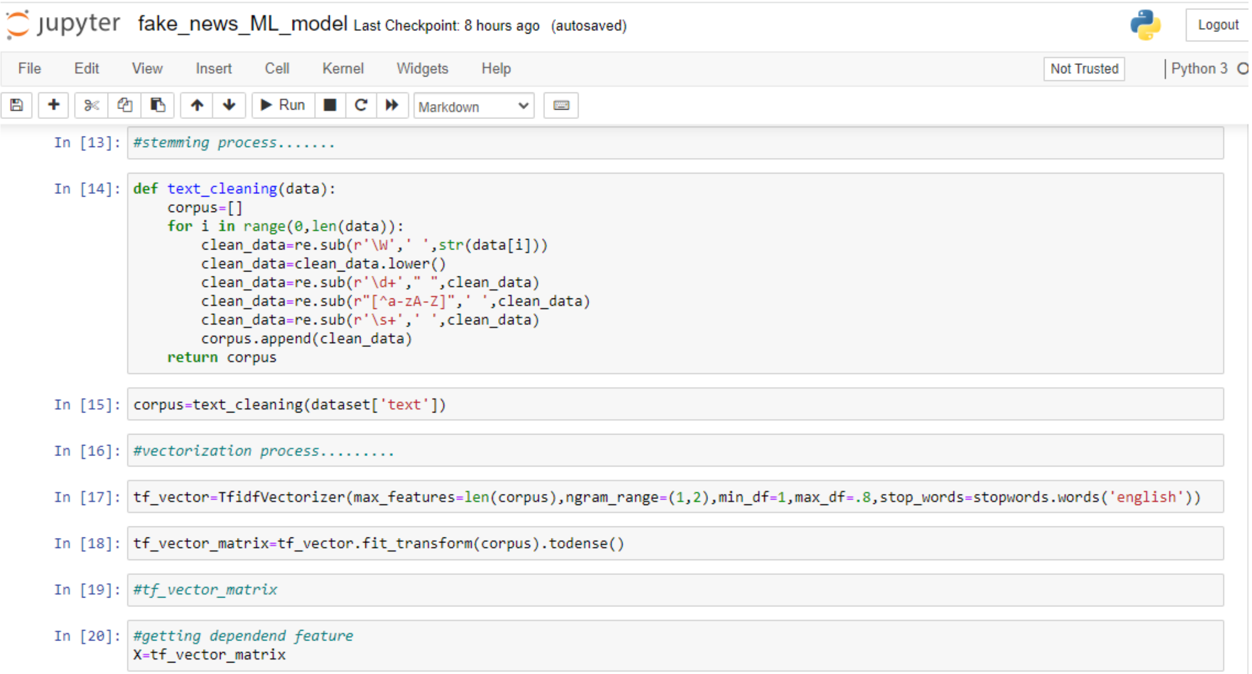
In [11]:
```
#dataset['text']=dataset['title']+ " " +dataset['text']
#this will delete all the other coloumns we do not need for the rest of the work
del dataset['title']
del dataset['author']
del dataset['id']
```

In [12]:
```
dataset.head(5)
```

Out[12]:

index	text	label

Cleaning the Data and vectorization using TF-IDF



In [13]:
```
#stemming process.......
```

In [14]:
```
def text_cleaning(data):
    corpus=[]
    for i in range(0,len(data)):
        clean_data=re.sub(r'\W',' ',str(data[i]))
        clean_data=clean_data.lower()
        clean_data=re.sub(r'\d+'," ",clean_data)
        clean_data=re.sub(r"[^a-zA-Z]",' ',clean_data)
        clean_data=re.sub(r'\s+',' ',clean_data)
        corpus.append(clean_data)
    return corpus
```

In [15]:
```
corpus=text_cleaning(dataset['text'])
```

In [16]:
```
#vectorization process.........
```

In [17]:
```
tf_vector=TfidfVectorizer(max_features=len(corpus),ngram_range=(1,2),min_df=1,max_df=.8,stop_words=stopwords.words('english'))
```

In [18]:
```
tf_vector_matrix=tf_vector.fit_transform(corpus).todense()
```

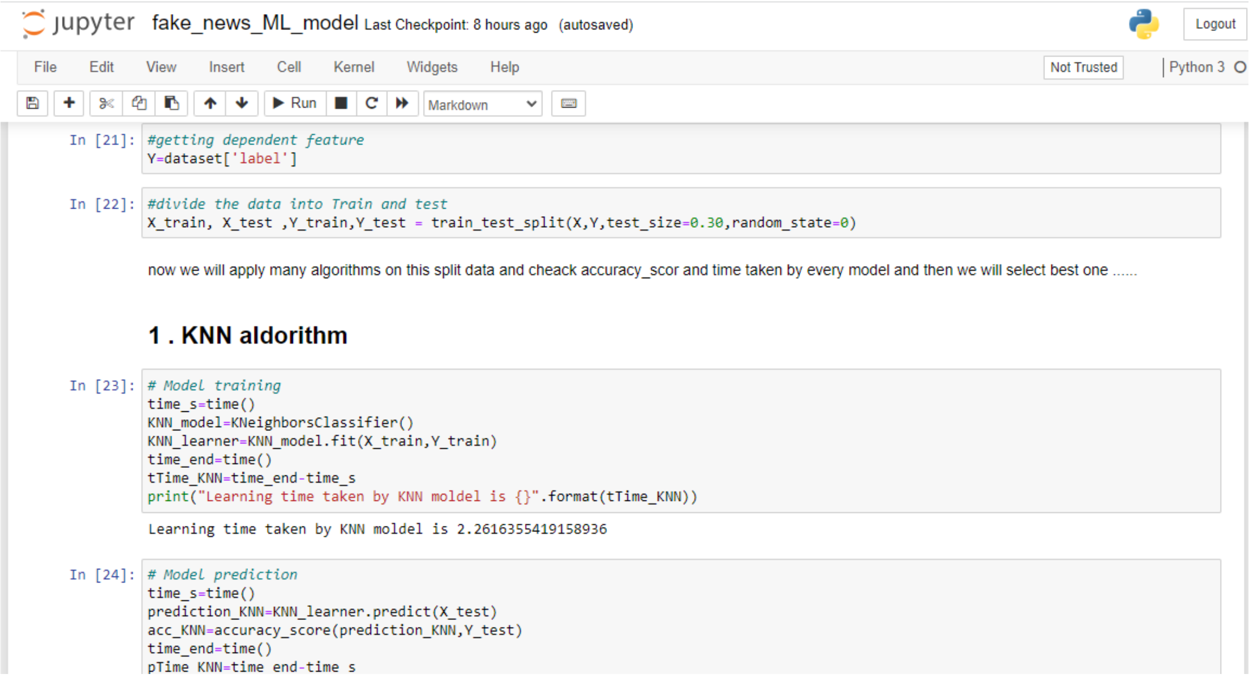
In [19]:
```
#tf_vector_matrix
```

In [20]:
```
#getting dependend feature
X=tf_vector_matrix
```

Applying the Algorithms

We applied the four algorithms →

1) KNN

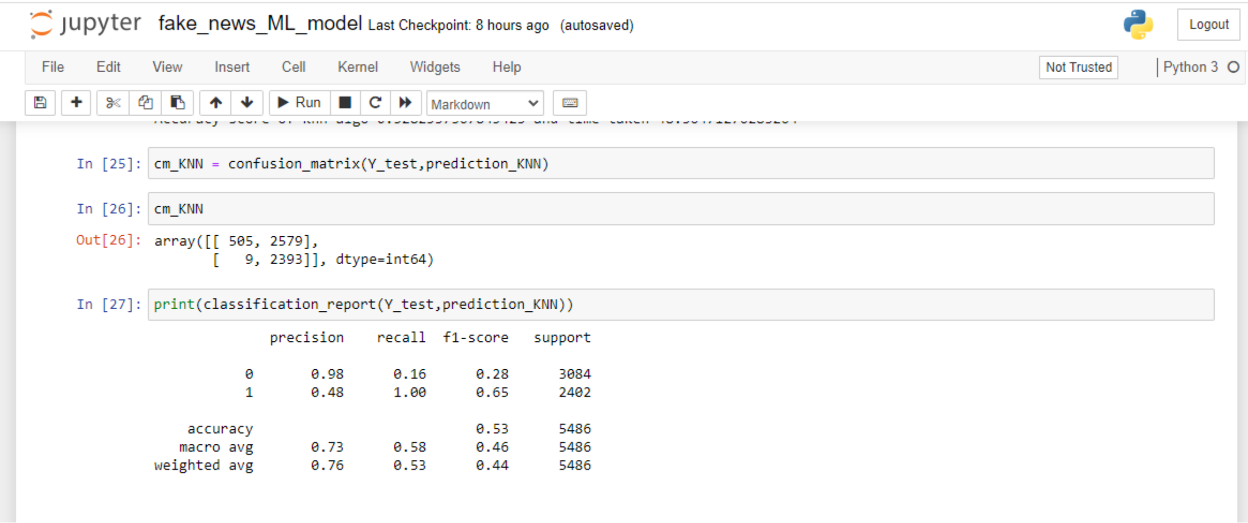


```
In [21]: #getting dependent feature
         Y=dataset['label']

In [22]: #divide the data into Train and test
         X_train, X_test ,Y_train,Y_test = train_test_split(X,Y,test_size=0.30,random_state=0)
```

now we will apply many algorithms on this split data and cheack accuracy_scor and time taken by every model and then we will select best one ......

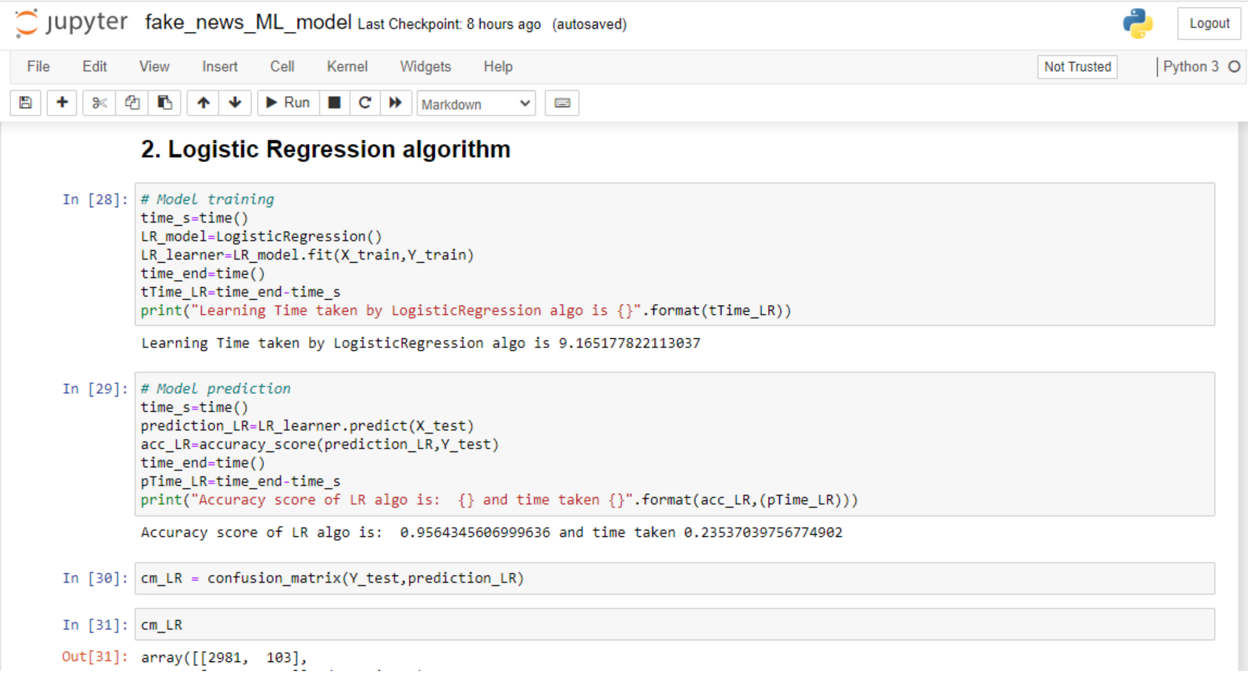
1 . KNN aldorithm

```
In [23]: # Model training
         time_s=time()
         KNN_model=KNeighborsClassifier()
         KNN_learner=KNN_model.fit(X_train,Y_train)
         time_end=time()
         tTime_KNN=time_end-time_s
         print("Learning time taken by KNN moldel is {}".format(tTime_KNN))

         Learning time taken by KNN moldel is 2.2616355419158936

In [24]: # Model prediction
         time_s=time()
         prediction_KNN=KNN_learner.predict(X_test)
         acc_KNN=accuracy_score(prediction_KNN,Y_test)
         time_end=time()
         pTime_KNN=time_end-time_s
```



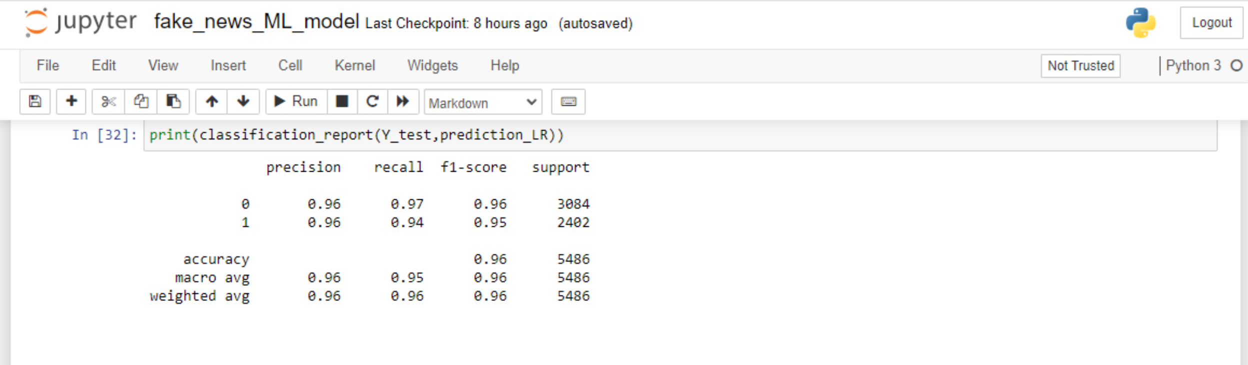
```
In [25]: cm_KNN = confusion_matrix(Y_test,prediction_KNN)

In [26]: cm_KNN

Out[26]: array([[ 505, 2579],
                [   9, 2393]], dtype=int64)

In [27]: print(classification_report(Y_test,prediction_KNN))

                       precision    recall  f1-score   support

                    0       0.98      0.16      0.28      3084
                    1       0.48      1.00      0.65      2402

             accuracy                           0.53      5486
            macro avg       0.73      0.58      0.46      5486
         weighted avg       0.76      0.53      0.44      5486
```

2) Logistic Regression algorithm



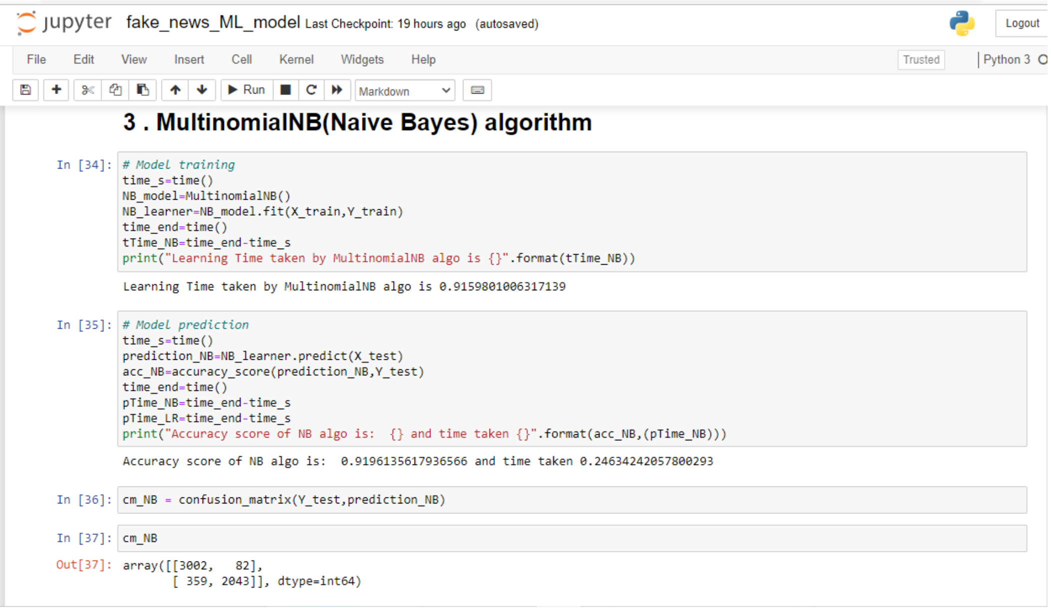
2. Logistic Regression algorithm

```
In [28]: # Model training
         time_s=time()
         LR_model=LogisticRegression()
         LR_learner=LR_model.fit(X_train,Y_train)
         time_end=time()
         tTime_LR=time_end-time_s
         print("Learning Time taken by LogisticRegression algo is {}".format(tTime_LR))

         Learning Time taken by LogisticRegression algo is 9.165177822113037

In [29]: # Model prediction
         time_s=time()
         prediction_LR=LR_learner.predict(X_test)
         acc_LR=accuracy_score(prediction_LR,Y_test)
         time_end=time()
         pTime_LR=time_end-time_s
         print("Accuracy score of LR algo is:  {} and time taken {}".format(acc_LR,(pTime_LR)))

         Accuracy score of LR algo is:  0.9564345606999636 and time taken 0.23537039756774902

In [30]: cm_LR = confusion_matrix(Y_test,prediction_LR)

In [31]: cm_LR

Out[31]: array([[2981,  103],
```



```
In [32]: print(classification_report(Y_test,prediction_LR))

                       precision    recall  f1-score   support

                    0       0.96      0.97      0.96      3084
                    1       0.96      0.94      0.95      2402

             accuracy                           0.96      5486
            macro avg       0.96      0.95      0.96      5486
         weighted avg       0.96      0.96      0.96      5486
```

3) Naive Bayes Algorithm (NB)


jupyter fake_news_ML_model Last Checkpoint: 19 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

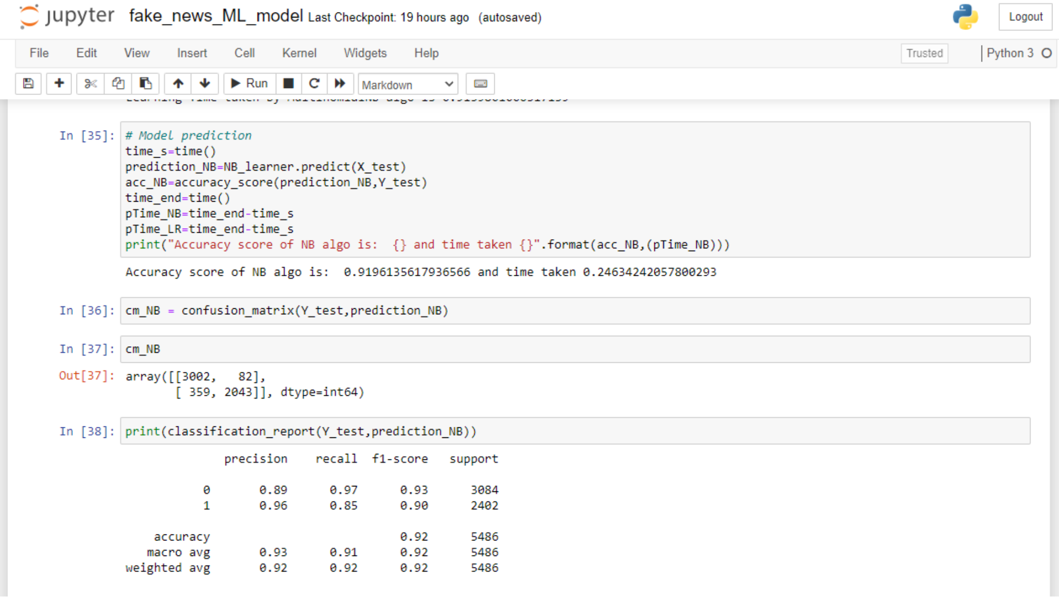
3 . MultinomialNB(Naive Bayes) algorithm

```
In [34]: # Model training
         time_s=time()
         NB_model=MultinomialNB()
         NB_learner=NB_model.fit(X_train,Y_train)
         time_end=time()
         tTime_NB=time_end-time_s
         print("Learning Time taken by MultinomialNB algo is {}".format(tTime_NB))

         Learning Time taken by MultinomialNB algo is 0.9159801006317139
```

```
In [35]: # Model prediction
         time_s=time()
         prediction_NB=NB_learner.predict(X_test)
         acc_NB=accuracy_score(prediction_NB,Y_test)
         time_end=time()
         pTime_NB=time_end-time_s
         pTime_LR=time_end-time_s
         print("Accuracy score of NB algo is:  {} and time taken {}".format(acc_NB,(pTime_NB)))

         Accuracy score of NB algo is:  0.9196135617936566 and time taken 0.24634242057800293
```

```
In [36]: cm_NB = confusion_matrix(Y_test,prediction_NB)
```

```
In [37]: cm_NB

Out[37]: array([[3002,   82],
                [ 359, 2043]], dtype=int64)
```


jupyter fake_news_ML_model Last Checkpoint: 19 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

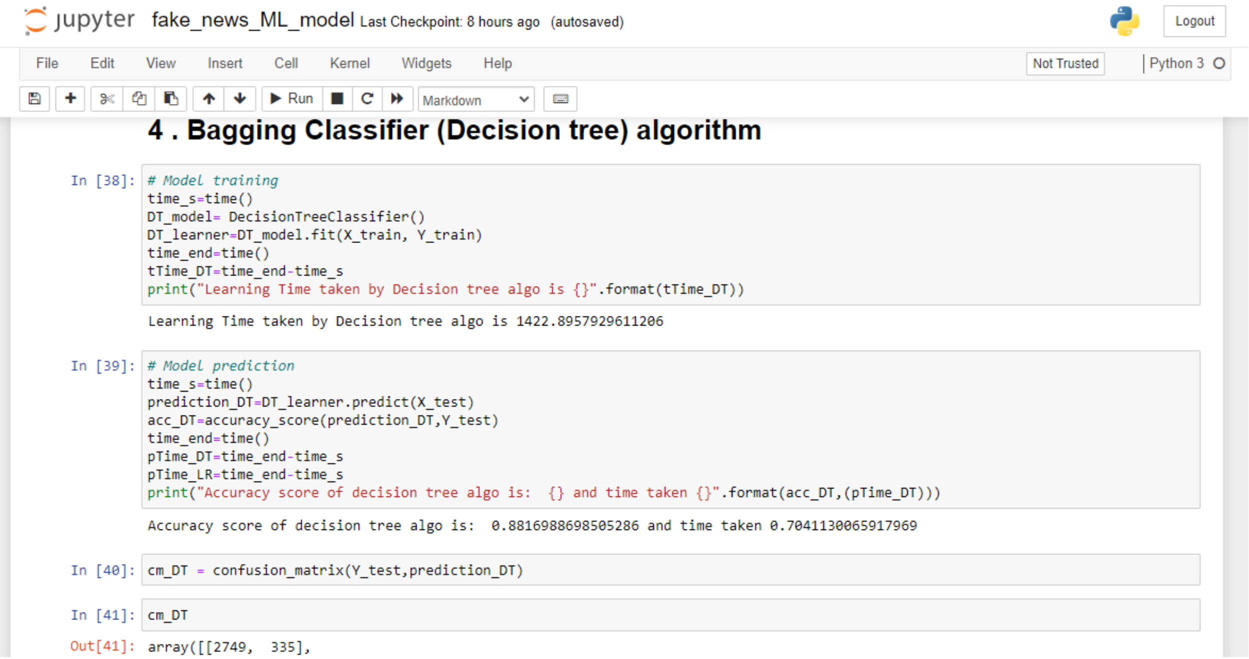
```
In [35]: # Model prediction
         time_s=time()
         prediction_NB=NB_learner.predict(X_test)
         acc_NB=accuracy_score(prediction_NB,Y_test)
         time_end=time()
         pTime_NB=time_end-time_s
         pTime_LR=time_end-time_s
         print("Accuracy score of NB algo is:  {} and time taken {}".format(acc_NB,(pTime_NB)))

         Accuracy score of NB algo is:  0.9196135617936566 and time taken 0.24634242057800293
```

```
In [36]: cm_NB = confusion_matrix(Y_test,prediction_NB)
```

```
In [37]: cm_NB

Out[37]: array([[3002,   82],
                [ 359, 2043]], dtype=int64)
```

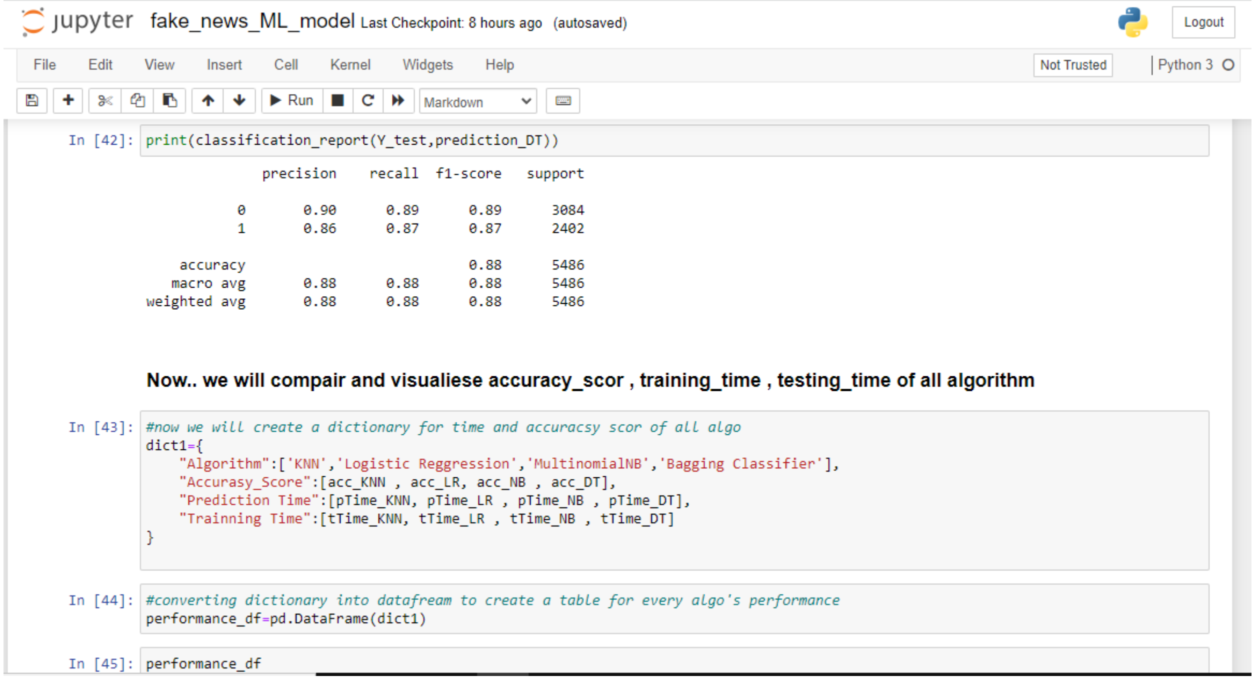
```
In [38]: print(classification_report(Y_test,prediction_NB))

                      precision    recall  f1-score   support

                   0       0.89      0.97      0.93      3084
                   1       0.96      0.85      0.90      2402

            accuracy                           0.92      5486
           macro avg       0.93      0.91      0.92      5486
        weighted avg       0.92      0.92      0.92      5486
```

4) Bagging classifiers



4 . Bagging Classifier (Decision tree) algorithm

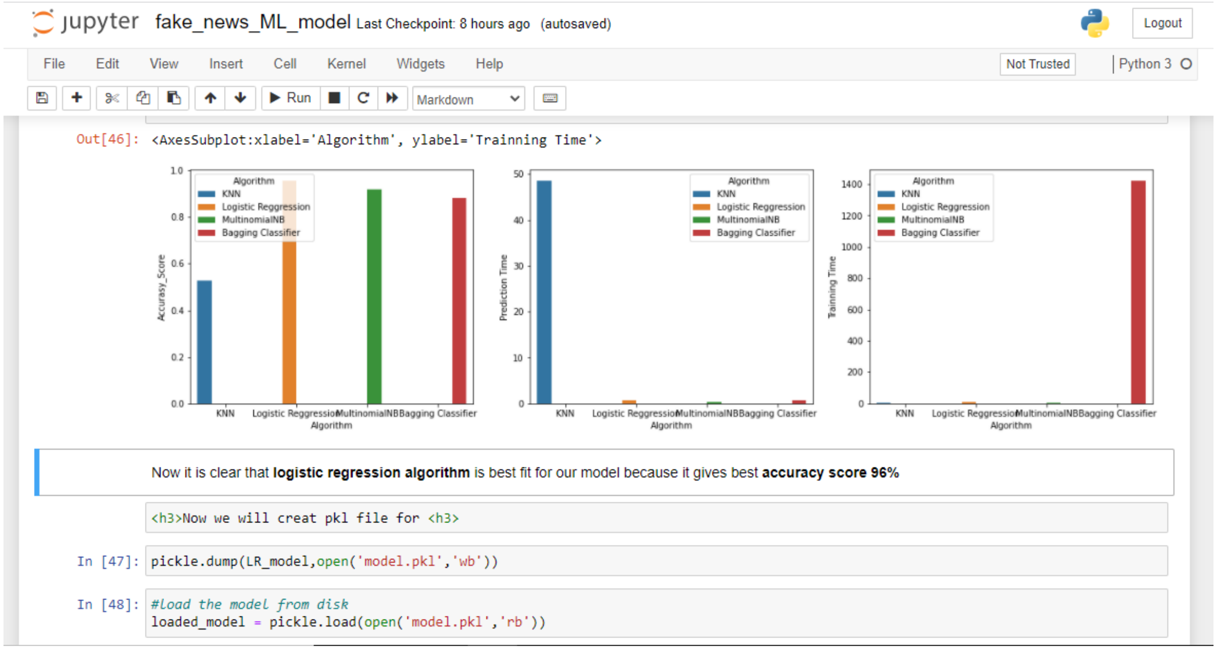
```
In [38]: # Model training
         time_s=time()
         DT_model= DecisionTreeClassifier()
         DT_learner=DT_model.fit(X_train, Y_train)
         time_end=time()
         tTime_DT=time_end-time_s
         print("Learning Time taken by Decision tree algo is {}".format(tTime_DT))

         Learning Time taken by Decision tree algo is 1422.8957929611206

In [39]: # Model prediction
         time_s=time()
         prediction_DT=DT_learner.predict(X_test)
         acc_DT=accuracy_score(prediction_DT,Y_test)
         time_end=time()
         pTime_DT=time_end-time_s
         pTime_LR=time_end-time_s
         print("Accuracy score of decision tree algo is:  {} and time taken {}".format(acc_DT,(pTime_DT)))

         Accuracy score of decision tree algo is:  0.8816988698505286 and time taken 0.7041130065917969

In [40]: cm_DT = confusion_matrix(Y_test,prediction_DT)

In [41]: cm_DT

Out[41]: array([[2749,  335],
```



```
In [42]: print(classification_report(Y_test,prediction_DT))

                       precision    recall  f1-score   support

                    0       0.90      0.89      0.89      3084
                    1       0.86      0.87      0.87      2402

             accuracy                           0.88      5486
            macro avg       0.88      0.88      0.88      5486
         weighted avg       0.88      0.88      0.88      5486
```

Now.. we will compair and visualiese accuracy_scor , training_time , testing_time of all algorithm

```
In [43]: #now we will create a dictionary for time and accuracsy scor of all algo
         dict1={
             "Algorithm":['KNN','Logistic Reggression','MultinomialNB','Bagging Classifier'],
             "Accurasy_Score":[acc_KNN , acc_LR, acc_NB , acc_DT],
             "Prediction Time":[pTime_KNN, pTime_LR , pTime_NB , pTime_DT],
             "Trainning Time":[tTime_KNN, tTime_LR , tTime_NB , tTime_DT]
         }

In [44]: #converting dictionary into datafream to create a table for every algo's performance
         performance_df=pd.DataFrame(dict1)

In [45]: performance_df
```



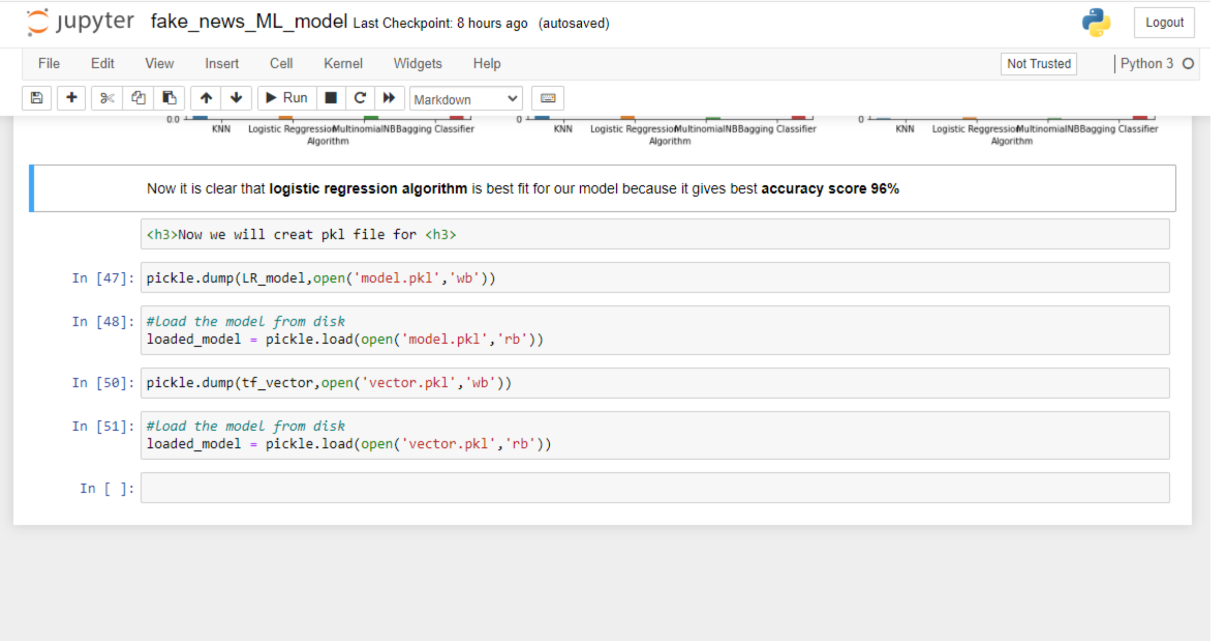
```
Out[46]: <AxesSubplot:xlabel='Algorithm', ylabel='Trainning Time'>
```

Now it is clear that **logistic regression algorithm** is best fit for our model because it gives best **accuracy score 96%**

```
<h3>Now we will creat pkl file for <h3>
```

```
In [47]: pickle.dump(LR_model,open('model.pkl','wb'))
```

```
In [48]: #Load the model from disk
         loaded_model = pickle.load(open('model.pkl','rb'))
```


jupyter
fake_news_ML_model Last Checkpoint: 8 hours ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3
Run
Markdown
0.0
KNN
Logistic ReggressionMultinomialNBBagging Classifier
Algorithm
0
KNN
Logistic ReggressionMultinomialNBBagging Classifier
Algorithm
0
KNN
Logistic ReggressionMultinomialNBBagging Classifier
Algorithm

Now it is clear that **logistic regression algorithm** is best fit for our model because it gives best **accuracy score 96%**

```
<h3>Now we will creat pkl file for <h3>
```

```
In [47]: pickle.dump(LR_model,open('model.pkl','wb'))
```

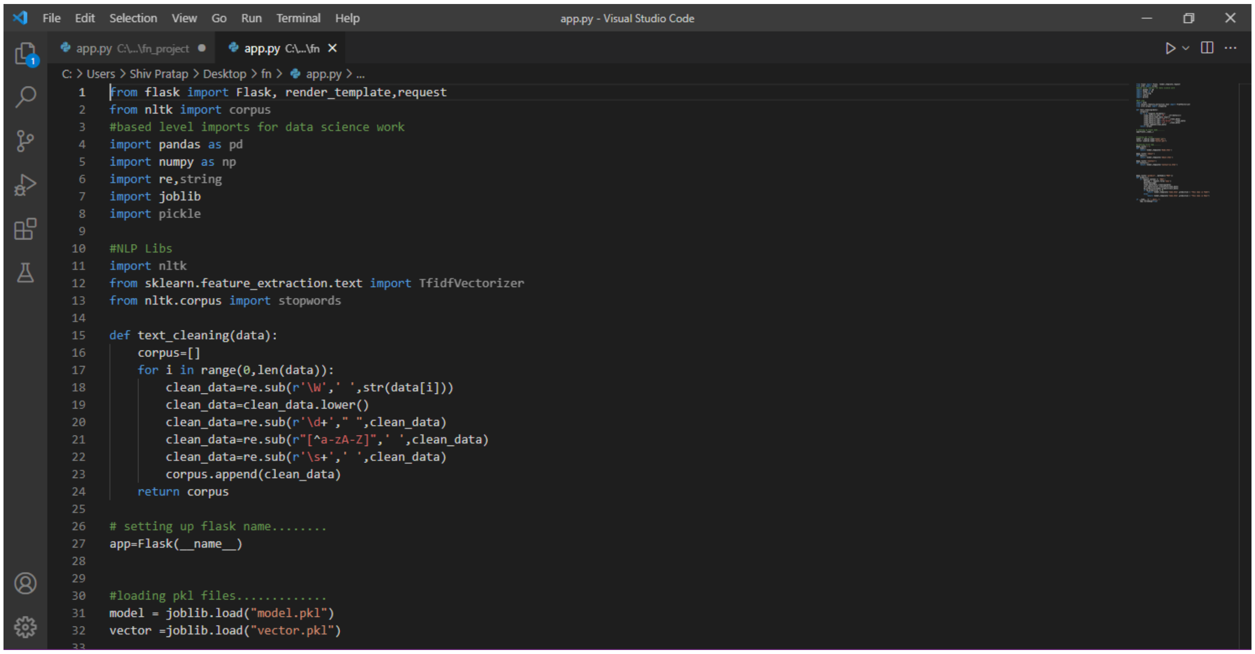
```
In [48]: #Load the model from disk
         loaded_model = pickle.load(open('model.pkl','rb'))
```

```
In [50]: pickle.dump(tf_vector,open('vector.pkl','wb'))
```

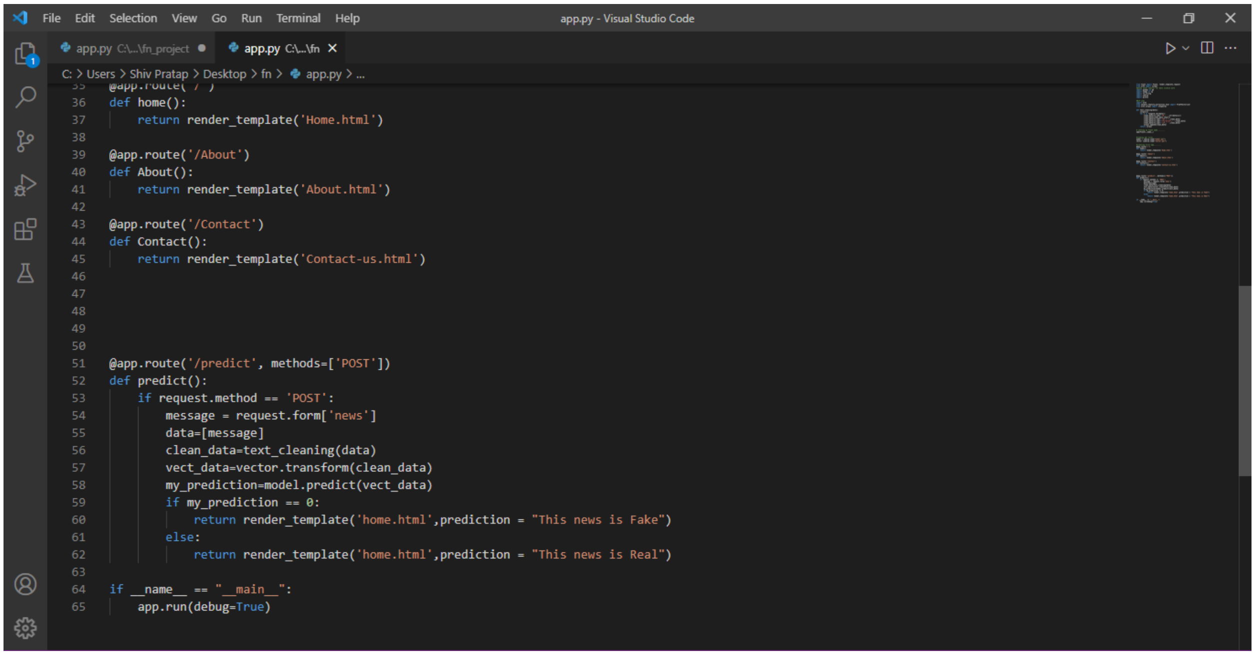
```
In [51]: #Load the model from disk
         loaded_model = pickle.load(open('vector.pkl','rb'))
```

```
In [ ]:
```

Flask implementation


File Edit Selection View Go Run Terminal Help
app.py - Visual Studio Code
app.py C:\...\fn_project
app.py C:\...\fn
C: > Users > Shiv Pratap > Desktop > fn > app.py > ...

```python
from flask import Flask, render_template,request
from nltk import corpus
#based level imports for data science work
import pandas as pd
import numpy as np
import re,string
import joblib
import pickle

#NLP Libs
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.corpus import stopwords

def text_cleaning(data):
    corpus=[]
    for i in range(0,len(data)):
        clean_data=re.sub(r'\W',' ',str(data[i]))
        clean_data=clean_data.lower()
        clean_data=re.sub(r'\d+'," ",clean_data)
        clean_data=re.sub(r"[^a-zA-Z]",' ',clean_data)
        clean_data=re.sub(r'\s+',' ',clean_data)
        corpus.append(clean_data)
    return corpus

# setting up flask name........
app=Flask(__name__)


#loading pkl files............
model = joblib.load("model.pkl")
vector =joblib.load("vector.pkl")
```


File Edit Selection View Go Run Terminal Help
app.py - Visual Studio Code
app.py C:\...\fn_project
app.py C:\...\fn
C: > Users > Shiv Pratap > Desktop > fn > app.py > ...

```python
@app.route('/')
def home():
    return render_template('Home.html')

@app.route('/About')
def About():
    return render_template('About.html')

@app.route('/Contact')
def Contact():
    return render_template('Contact-us.html')





@app.route('/predict', methods=['POST'])
def predict():
    if request.method == 'POST':
        message = request.form['news']
        data=[message]
        clean_data=text_cleaning(data)
        vect_data=vector.transform(clean_data)
        my_prediction=model.predict(vect_data)
        if my_prediction == 0:
            return render_template('home.html',prediction = "This news is Fake")
        else:
            return render_template('home.html',prediction = "This news is Real")

if __name__ == "__main__":
    app.run(debug=True)
```

Full code on GitHub Repository Link:

Deployment

We deployed our model on Heroku. Link:

Real world implementation

To implement this in real life, we can make a mobile app/website or a whatsapp-integrated feature.

Users would simply enter the link of a news article/ website and be able to verify whether a news is real or fake.

Conclusion

Our data have (20800,5) items and We used to test KNN (k-nearest neighbors), Logistic regression, Bagging classifier, Naive Bayes Algorithm (NB) for Fake News Detection Model Using NLP &Machine Learning

Accuracy score of KNN is 52% and Prediction time is 77.70

Accuracy score of Logistic regression is 95% and Prediction time is 0.33

Accuracy score of Bagging classifier is 88% and Prediction time is 0.62

Accuracy score of Naive Bayes Algorithm (NB) is 91% and Prediction time is 0.35

So overall, the performance for our dataset was better with the "Logistic Regression" Algorithm so we have selected this for our model. The confusion matrix has been plotted and the accuracy score has been measured for the performance analysis purposes.

Our systems take input from an URL or an existing database and classify it to be true or fake. To implement this, various NLP and Machine Learning Techniques have to be used.

Reference:

Data source: **https://www.kaggle.com/** | **https://www.kaggle.com/c/fake-news/data**